WEST

Generate Collection | Print

L3: Entry 32 of 39 File: USPT Apr 24, 1984

DOCUMENT-IDENTIFIER: US 4445174 A
TITLE: Multiprocessing system including a shared cache

Detailed Description Text (15):
A processor memory request predetermined sequence of operation will be described for processor A, with the understanding that a memory request sequence for processor B is similar. A serial predetermined sequence of interrogation or memory request is set forth for ease of explanation. The sequence is to interrogate CPU A's private cache, than the shared cache, than cross interrogate CPU B's private cache and last interrogate main memory. It is to be appreciated, that other predetermined sequences of interrogation are contemplated in the practice of the present invention. For example, the sequence may start with the concurrent interrogation of CPU A's private cache and the shared cache, with the sequence than proceeding as set forth above. A processor A memory request is first provided via line 16 to cache control and directory 20 for private cache A. The type of request, either a fetch or a store, is indicated by appropriate control signals on the line 16. If the target word is resident in the cache 8, as indicated by a directory address compare and a valid bit equal to ONE the cache control logic and directory 20 signals processor A accordingly, and the fetch or store takes place at cache A. The requested data is then either provided to or from processor A depending upon whether the request was a fetch or store.

Detailed Description Text (25):
Refer now to FIGS. 4.1 and 4.2 which set forth the four way set associative shared cache, directory and update/replacement array and logic. The logic set forth operates in a manner similar to that for a private cache as set forth in FIG. 2, with the difference being that a CPU priority logic network 75 is needed to determine which processor or the SCU receives priority when multiple requests occur. Store or fetch requests from processor A, processor B and SCU 38 are provided on lines 76, 78 and 80 to the inputs of the CPU priority logic network 75. The logic network 75 is a standard type logic network which provides a select output signal on output lines 82, 84 or 86, dependent upon which of the requesting devices provide the first request. In the event there are concurrent requests, the highest priority unit will be selected.

WEST

Generate Collection | Print

L3: Entry 26 of 39 File: USPT Jan 27, 1998

DOCUMENT-IDENTIFIER: US 5713004 A
TITLE: Cache control for use in a multiprocessor to prevent data from ping-ponging between caches

Brief Summary Text (17):
U.S. Pat. No. 4,445,174 discloses a shared cache that adds a shared cache in parallel with the private caches. That approach does not work for commodity microprocessors where the cache architecture is determined by the chip vendor. U.S. Pat. No. 4,484,267 teaches turning a write-back cache (called a store-in-cache SIC in that patent) into a write-through cache (called a store-through ST cache in that patent). In the configuration addressed by the current invention the shared caches are always write-through so the teaching of U.S. Pat. No. 4,484,267 does not apply.

WEST

Generate Collection Print

L3: Entry 23 of 39 File: USPT Apr 27, 1999

DOCUMENT-IDENTIFIER: US 5898849 A
TITLE: Microprocessor employing local caches for functional units to store memory operands used by the functional units

Detailed Description Text (5):
In parallel with conveying the address corresponding to a memory operand to the corresponding local cache 15A-15E, the functional units convey the address to global tags and control unit 13. Global tags and control unit 13 stores the tags for the cache lines stored in each of local caches 15A-15E. Additionally, each of local caches 15A-15E stores tags for the cache lines stored in that local cache 15A-15E. If a hit is detected for a requested memory operand in the local cache 15A-15E connected to the requesting functional unit, then the data is provided rapidly to the requesting functional unit. The functional unit uses the data provided by the local cache 15A-15E if a hit is detected. If a miss is detected in the local cache 15A-15E connected to the requesting functional unit, but global tags and control unit 13 detects that the memory operand hits in a different local cache 15A-15E or central data cache 14 (a "remote cache"), then the memory operand is forwarded from the different local cache 15A-15E or central data cache 14 to the requesting functional unit. Additionally, the cache line containing the memory operand is transferred into the local cache 15A-15E and is invalidated in the cache which provides the cache line. Finally, if global tags and control unit 13 detects a miss in all caches, global tags and control unit 13 initiates a transfer of the accessed cache line from the main memory subsystem. The cache line is stored into the local cache 15A-15E corresponding to the requesting functional unit.

WEST

Generate Collection Print

L3: Entry 33 of 39 File: USPT Apr 10, 1984

DOCUMENT-IDENTIFIER: US 4442487 A
TITLE: Three level memory hierarchy using write and share flags

Detailed Description Text (49):
FIG. 6.1 and FIG. 6.2 comprise a block diagram illustrating how a single processor in this instance processor A, accesses its private and shared caches. A memory request is initiated by sending a memory request and address to both the private and shared L1 cache directories. If there is an address compare at either directory and the line is valid (V=1), then the associated L1 cache is accessed. Assuming a four-way set associative cache, four words are read out in parallel from the cache and the requested word is gated to or from the processor.

Detailed Description Text (51):
Both caches at L1 or L2 can be accessed in parallel because a line or page is either in the private caches or shared caches but never in both simultaneously.

WEST

Generate Collection Print

L3: Entry 21 of 39 File: USPT Jan 11, 2000

DOCUMENT-IDENTIFIER: US 6014756 A
TITLE: High availability error self-recovering shared cache for multiprocessor systems

Brief Summary Text (14):
If, for example, a processing unit requests a line which is not available in its private L2 cache, then this "request for a missing line" is routed to the shared cache and to the main memory in parallel. If the shared cache can provide the requested data, the request to the main memory will be cancelled. The main memory provides the data only in those cases where no match is found in the shared cache directory. Data which is provided by the main memory is loaded into the shared cache and the private cache as well. Any further accesses to this cache line by any other processing unit are then served by the shared cache again to improve the performance of the overall system as above described.

Detailed Description Text (29):
In FIG. 4 the functionality of the shared cache is illustrated by another example, where a linefetch request with a parity error in the shared cache directory or the LRU logic occurs. In the first cycle a linefetch request is sent to the shared cache and to the main memory. The requested line is found in the cache directory, but a parity error is detected in the addressed congruence class. Therefore, in the next cycle the memory request for data is not cancelled ("*" and dotted line in FIG. 4) due to the parity error in the shared cache directory or the LRU. Instead of cancelling, at the same cycle all cache directory entries in the addressed congruence class are invalidated by setting the corresponding Valid bits to "0". Some cycles later the main memory serves the linefetch request by the processing unit (FIG. 4 "**"), wherein data blocks "0"-"N" are loaded in parallel into the shared cache and the private cache associated with the requesting processing unit. In the cycle where the last block "N" of data has been transferred to the shared cache, this line is validated in the shared cache by setting the Valid bit to "1" in the corresponding entry of the shared cache directory.

Parallel Caching

WEST

Generate Collection Print

L3: Entry 13 of 39 File: USPT Aug 15, 2000

DOCUMENT-IDENTIFIER: US 6105109 A
TITLE: System speed loading of a writable cache code array

Detailed Description Text (4):
FIG. 1. The S/390 G4 memory hierarchy structure for these modern symmetrical multiprocessors (SMPs) uses a shared L2 cache chip where cache controls are on the same chip, and the shared cache has two chips clustered as the L2 cache pair with the entire memory address mapped across the pair of L2 cache chips, and these are interleaved across four memory cards 0,1,2,3 to maximize concurrent operation. The cache array is a six-way set-associative, dual inter-leave cache with directories on the chips dedicated to each system bus 0, 1, 2, 3. Each L2 chip itself has nonblocking switching to each attached CP of the SMP as well as to other system components. The independent cross point switches maintain the SMP performance and operate independently for each CP and BSN port and for each cache array, allowing up to five simultanteous transfers on the L2 cache chip data ports. Data buffers in the cache array minimize latency. Each CP chip has on board the L1 cache designed as a store through cache, meaning that altered data is also stored back to the L2 next level cache when the data is altered. The L2 however is a store-in. Data coherency between the Lis and L2s is maintained so that all lines contained in any L1 are also stored in that cluster's L2. The cache has ECC on both the cache and directory arrays. Error correction is in all cases done on the fly. This means that when uncorrectable errors are encountered, there must be system recovery. In addition, on the BSN chip there are L2.5 caches as the third level of the memory hierarchy located on each of the four logical BSN buses. These operate as a main store cache for frequently accessed, shared, read only data and as a store through cache. There are no superset or subset rules between L2 and L2.5 and L2.5 does not hold data that is more recent than the copy in either the L2 or main store. But also, the hierarchy which makes the L2 caches supersets of the L1 cache of the same cluster and means that the L1 line must also exist at the L2 level of the same cluster, but the reverse is not true normally, in that a line of data may exist in the L2 without existing in the L1 cache, except for a subset rule relating to what is called in the S/390 structure millicode. The S/390 millicode has included nonupdatable millicode which is kept in the L1 when not in the L2 in order to prevent certain deadlocks on the BSN bus from occuring. Coherency is maintained by use of the directory states that are defined for the first two levels of caches.